B. TECH. PROJECT REPORT

On

Human hand segmentation using shape prior for bio-metrics

BY

Chaparala Hari Kishore

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

Human Hand Segmentation using shape prior for bio-metrics

A PROJECT REPORT

Submitted in partial fulfillment of the

requirements for the award of the degrees

of

BACHELOR OF TECHNOLOGY

in

ELECTRICAL ENGINEERING

Submitted by:

Chaparala Hari Kishore

Guided by:

Dr. Vivek Kanhangad

INDIAN INSTITUTE OF TECHNOLOGY INDORE

December 5th 2016

CANDIDATE'S DECLARATION

I hereby declare that the project entitled **"Human hand segmentation using shape prior for bio-metrics"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Electrical Engineering Department' completed under the supervision of **Dr. Vivek Kanhangad,** IIT Indore is an authentic work.

Further, I declare that I have not submitted this work for the award of any other degree elsewhere.

Signature and name of the student with date

CERTIFICATE by BTP Guide

It is certified that the above statement made by the student is correct to the best of my knowledge.

Signature of BTP Guide with dates and designation

Preface

This report on “Human Hand segmentation using shape prior for bio-metrics" is prepared under the guidance of Dr. Vivek Kanhangad.

This report mainly focuses on segmentation of human hand with shape and texture knowledge and how my algorithm can give good results in challenging backgrounds with clutter and varying illumination where most of the existing approaches cannot give promising results.

I have tried to the best of my abilities and knowledge to explain the content of my project in a lucid manner.

Chaparala Hari Kishore
B.Tech. IV Year
Discipline of Electrical Engineering
IIT Indore

Acknowledgements

I have taken efforts in this project. However, it would not have been possible without the kind support and help of Dr. Kanhangad and IIT Indore.

I am highly indebted to Dr. Kanhangad for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.
I would like to express my gratitude towards my parents for their kind co-operation and encouragement which helped me in completion of this project.
I would like to express my special gratitude and thanks to Electrical engineering department for giving me such attention and time.

Also I would like to thank all the developers of OpenCV, an open source project for computer vision and image processing applications.

Chaparala Hari Kishore

B.Tech. IV Year

Discipline of Electrical Engineering

IIT Indore

Abstract

In this project an improved and robust hand segmentation method is presented for cluttered backgrounds with significant illumination changes. A combination of Active Appearance Model, skin color model and Haar cascade classifier is used for this purpose with an novel algorithm for the improvisation of initialization of the Appearance model learned from the images. The resulting approach is able to segment human hand in challenging backgrounds making it robust to most of the real life applications.

Table of Contents

List of Figures

Chapter 1

Introduction

Segmentation is the process of separating the image into background and foreground based on the region of interest which will be our foreground. Most of the capturing devices for hand involves the use of pegs or require the user to place the hand in a controlled environment making them less user friendly and they are hygiene issues when it comes to contact based approaches. This poses a major problem and this can be done away using a low resolution camera as the hand geometry and palm print based recognition systems don't need high definition captures for recognition. And they present no hygiene issues. Sometimes it is necessary to segment all the regions of hand rather than just the palm based on the method of recognition. Skin color is the most used modality for hand segmentation. However it is very susceptible to luminosity changes.

Segmentation of any object from background has some major hindrances namely occlusion, clutter, illumination and deformation. In the case of hand segmentation, illumination, background clutter and deformation are the major ones. So to address these issues a robust approach must be designed which can segment human hand in cluttered backgrounds, varied deformations by using minimal information of skin color to challenge illumination changes. Shape based segmentation techniques present a solution to these problems and can segment the hand robustly. There are various shape based approaches in literature that can be implemented for hand segmentation. Shape contexts trained with SVM that can be used to segment the hand by prepossessing the image with hierarchical segmentation and looking for the region which most resembles the hand. But this is time consuming and cannot give accurate results as it converges to the nearest segment making it susceptible to miss minor regions of the hand. Other approaches like level sets or graph cut based shape prior segmentation can be used. Though these promise accurate results but cannot account to all the deformations of the hand and are very time consuming as they have to match the segment at every iteration to all the shape templates. Doublet at [3] proposed a way of segmenting the human hand using the Active Shape Modeling(ASM) by cootes at [7]. The method used in [3] is by using skin color modeling and gradient analysis to separate the hand. However this assumes that all the pixels whose gradient is greater than a predefined value as background pixels and it also faces a major drawback when skin is not segmented properly. Also the initialization of the learned Active Shape Model(ASM) is done based on the skin color segmentation results. However, if some regions of the hand are missing after skin segmentation this may lead to improper initialization and divergence of the ASM. In order to face these issues I propose the segmentation using AAM which is more robust than ASM and has lesser problems with initialization. Also I propose a novel algorithm for the initialization of the shape prior which can face most of the cases where skin color segmentation is not done properly. More over I am using a combination of a classifier and skin color modeling for the detection of hand in the image which has an advantage over gradient and skin color based hand detection at [3] especially under varying illumination.

Chapter 2

Hand Detection

I have divided the entire segmentation process into detection phase and Shape based segmentation phase.

Hand detection: Hand detection is performed using a combination of **Haar cascade classifier** and **Skin color modeling** to remove unnecessary false positives as the classifier is susceptible to few false detections for hand.

2.1 **Haar Cascade classifier**

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Initially, the algorithm needs a lot of positive images (images of hand) and negative images (images without hand) to train the classifier. Then we need to extract features from it. For this, Haar features shown in below (Fig 1) image are used. They are just like convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

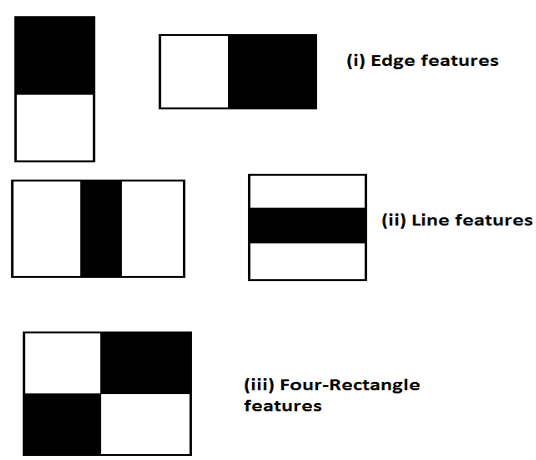

Figure 1: Various Haar features

Now all possible sizes and locations of each kernel is used to calculate plenty of features. (Just imagine how much computation it needs? Even a 24x24 window results over 160000 features). For each feature calculation, we need to find sum of pixels under white and black rectangles. To solve this, the authors [10] introduced the integral images. It simplifies calculation of sum of pixels, how large may be the number of pixels, to an operation involving just four pixels. This makes things super-fast!

But among all these features, most of them are irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost [11]. It can be used in conjunction with many other types of learning algorithms to improve their performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing (e.g., their error rate is smaller than 0.5 for binary classification), the final model can be proven to converge to a strong learner.

For this, I applied each and every feature on all the training images. For each feature, it finds the best threshold which will classify the hand to positive and negative. But obviously, there will be errors or misclassifications. I have selected the features with minimum error rate, which means they are the features that best classifies the hand and non-hand images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then again same process is done. New error rates are calculated. Also new weights. The process is continued until required accuracy or error rate is achieved or required number of features are found).

Final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others(Adaboost in this case) forms a strong classifier. The paper [10] says even 200 features provide detection with 95% accuracy. The final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. Which is a big gain)

So now I take an image. Take each 24x24 window. Apply 6000 features to it. Check if it is hand or not. But, Isn't it a little inefficient and time consuming? Yes, it is. Authors have a good solution for that.

In an image, most of the image region is non-hand region. So it is a better idea to have a simple method to check if a window is not a hand region. If it is not, discard it in a single shot. Don't process it again. Instead focus on region where there can be a hand. This way, we can find more time to check a possible hand region.

For this the concept of **Cascade of Classifiers** is used. Instead of applying all the 6000 features on a window, group the features into different stages of classifiers and apply one-by-one. (Normally first few stages will contain very less number of features). If a window fails the first stage, discard it. We don't consider remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a hand region.

I have taken 6000+ features with 38 stages with 1, 10, 25, 25 and 50 features in first five stages. (Two features in the above image is actually obtained as the best two features from Adaboost). On an average, 10 features out of 6000+ are evaluated per sub-window.

Integral images can be defined as two-dimensional lookup tables in the form of a matrix with the same size of the original image. Each element of the integral image contains the sum of all pixels located on the up-left region of the original image (in relation to the element's position). This allows to compute sum of rectangular areas in the image, at any position or scale, using only four lookups:

sum= I(A)+I(B)-I(C)-I(D)

Where A,B,C and D belong to the integral of the image I, as shown below.

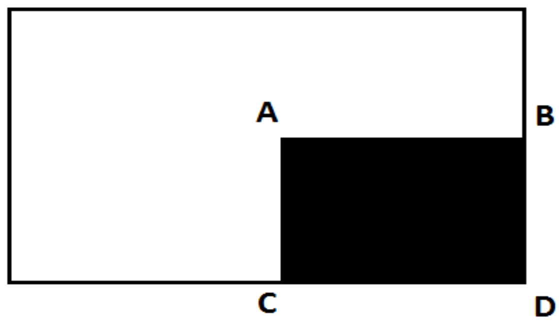

Figure 2

A set of 900 images of hand and 2000 non-hand images are used for the classifier training. The classifier is susceptible to some false positives which can be dealt with using the approach in the next section.

2.2 **Skin color segmentation** Skin color modeling is done using the same approach as of [3] using neural networks which gave faster and better results.

Classical skin distribution models are defined assuming some restrictions. In one hand, models representing skin color without generalization must completely describe all skin and background tones, that it is non-realistic in practice. In the other hand, parametric models like Gaussian Mixture Model suppose skin color distribution can be mathematically modeling. Without assuming these constraints, machine learning can model skin tone. I have used two neural network configurations to model skin tone. The first one contains one hidden layer with N ∈ [,1 20] neurons and it has a non linear transfer function while the second one has a linear transfer function with N ∈ [1, 10] neurons. The inputs of each network are the pixel's color components in YCBCR color domain. The output is composed by one neuron whose the value s ∈ [1,0] corresponds to the probability that a color is a skin color. For binary segmentation I have taken the threshold probability as 0.5.

The figure below shows the Neural Network used.

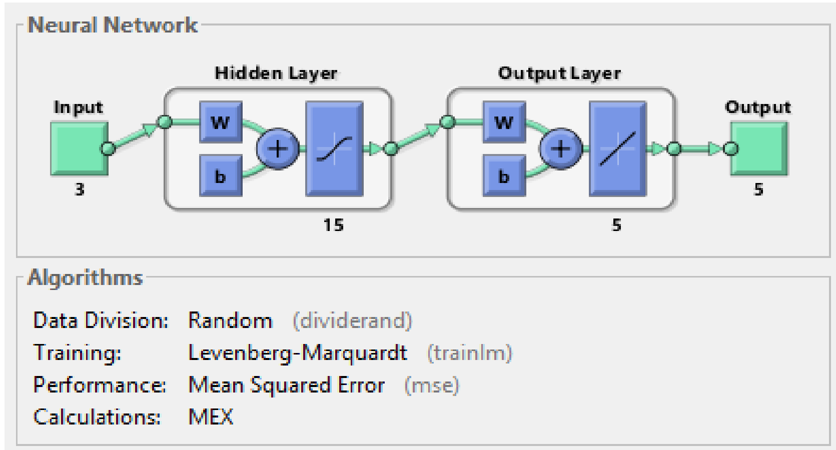

Figure 3: Neural Network

The Performance and regression plots are given below

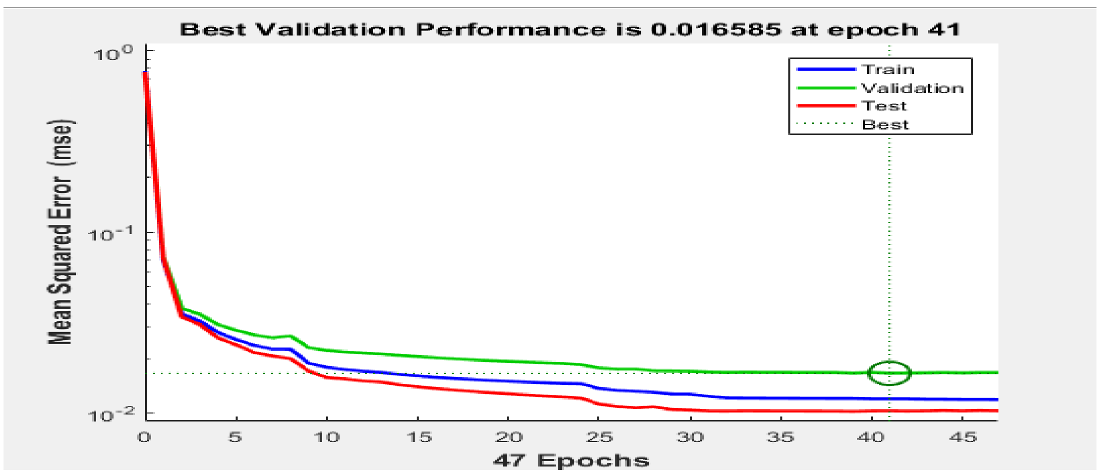

Figure 4: Performance Plot

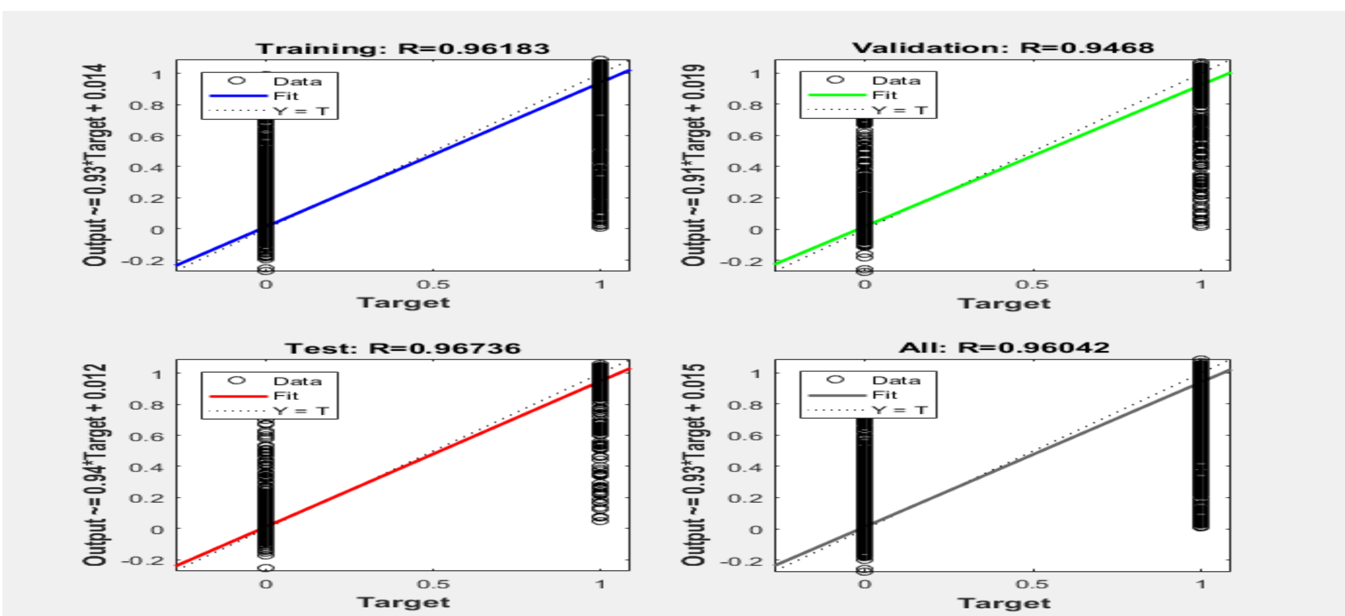

Figure 5: Regression Plots

2.3 Comparison of Neural Networks and Gaussian Mixture

The below table gives the performance results of most used method for classification. Though SVM's are giving better results but the execution time is pretty high compared to others.

	GAUSSIAN	NN	SVM
TIME	764ms	2626ms	69097ms
ERROR	13.6%	6.2%	4.9%

Figure 6: Comparative analysis [3]

The below figures demonstrate the performance of NN vs Gaussian Mixtures.

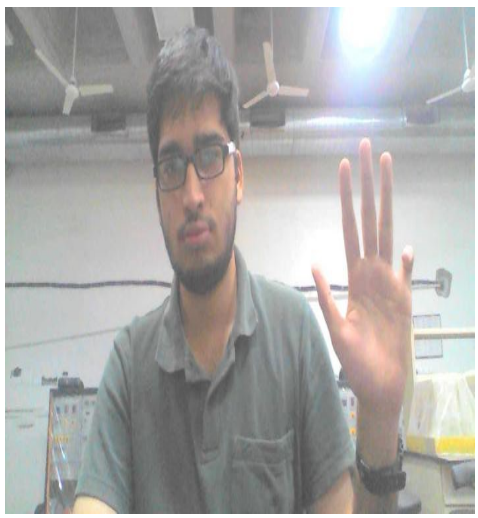

Figure 7: Test Image 1

Figure 8: Gaussian Test 1

Figure 9: Neural Networks Test 1

2.4 Skin color segmentation results in challenging and normal backgrounds

Shown below is the comparison between segmentation results in challenging and normal backgrounds.

Figure10: Test image 2 Normal Back ground

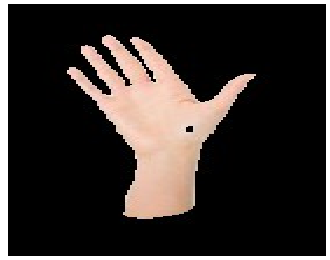

Figure 11: results Test 2

Figure 12: Test image 3 Normal back ground

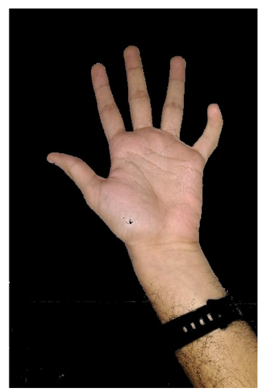

Figure13: Results Test 3

Shown Below are the segmentation results in a challenging background and it is clear from them that why skin color segmentation cannot give good results alone.

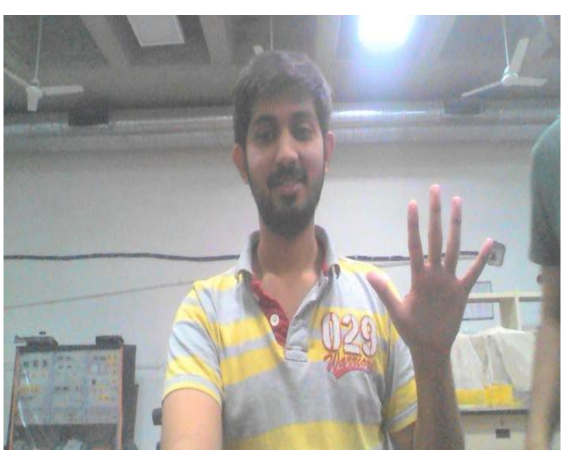

Figure 14: Test Image 4

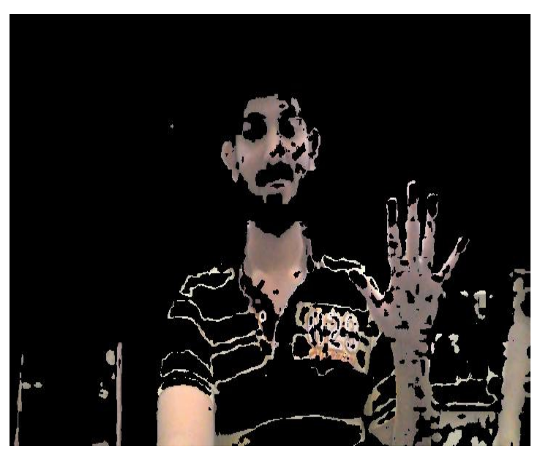

Figure15: Skin color Segmentation using NN's Test image 4

2.5 Detection results after using both Haar classifier and skin color segmentation

Figures 16 and 17 show the results after false detection are removed. Figures 18 and 19 show few results with no false detections. Note that the trained Cascade classier is used on the image with face region subtracted if any, using Viola- Jones algorithm.

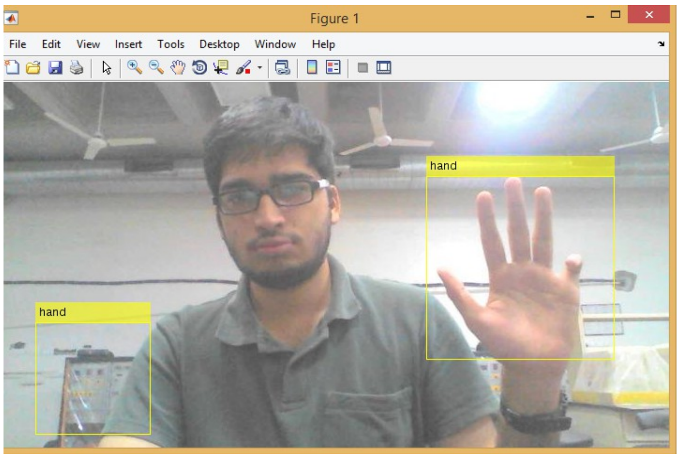

Figure 16: False detection to the left Test imgae 1

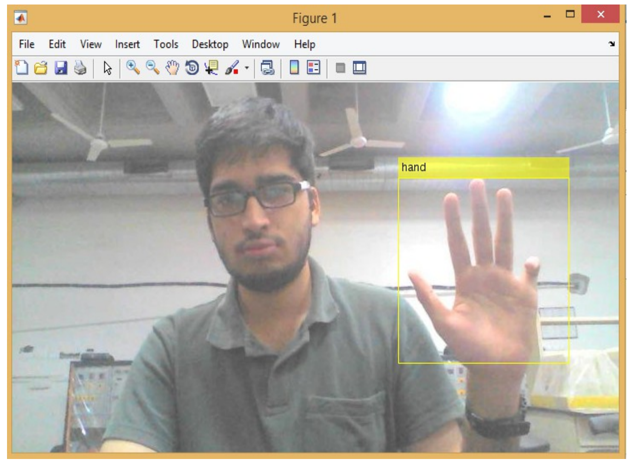

Figure 17: false detection removed using skin color Test image 1

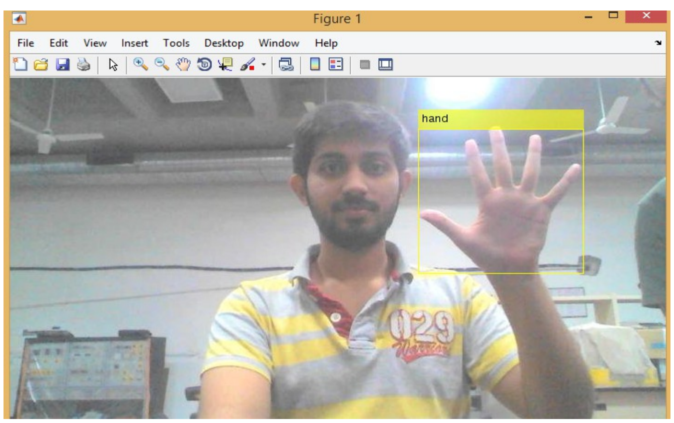

Figure 18: Hand Detection in Test image 5

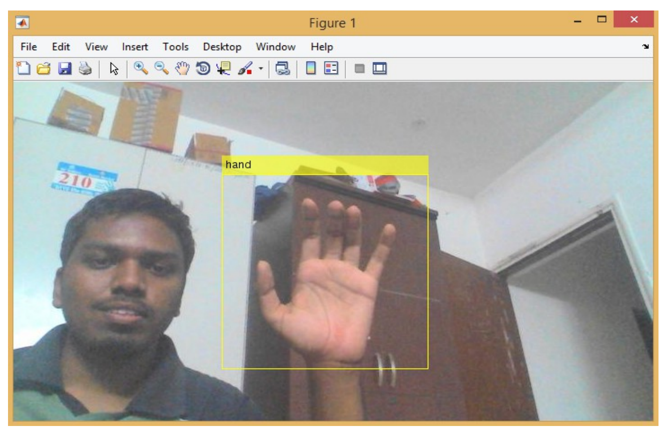

Figure 19: Hand detection in Test image 6

Now that our Hand is detected the next step is to segment it using Shape Prior. For this a modified version of Active Appearance Model[7] is used which is described is the next chapter.

Chapter 3

Active Appearance Model

ASM achieves faster feature point location than the AAM [12]. However, as it explicitly minimizes texture errors and increases the accuracy, the AAM gives a better match to the image texture. AAM is a a more generalized approach and as for the hand both the shape and boundary play an important role, AAM is used for my project.

A comparative study can be obtained from [13] which is verified in the project and ASM's are very prone to convergence issues compared to AAM's but are faster in detection.

Active Appearance Model (AAM) is a statistical deformable model of the shape and appearance of a deformable object class. It is a generative model which during fitting aims to recover a parametric description of a certain object through optimization.

A shape instance of a deformable object is represented as $S=[x_1, y_1, \ldots\ldots, x_L, y_L]^T$, a 12L×1 vector consisting of L landmark points coordinates $(x_i, y_i) \forall i=1,2,\ldots,L$. An AAM is trained using a set of N images $(I_1, I_2, \ldots, I_N)$ that are annotated with a set of L landmarks and it consists of the following parts:

3.1 Shape Model

The training shapes $(s_1, s_2, \ldots, s_N)$ are first aligned using Generalized Procrustes Analysis and then an orthonormal basis is created using Principal Component Analysis (PCA) which is further augmented with four eigenvectors that represent the similarity transform (scaling, in-plane rotation and translation). This results in $(\bar{s}, U_s)$ where $U_s \in \Re^{2L \times n}$ is the orthonormal basis of n eigenvectors (including the four similarity components) and $\bar{s} \in \Re^{2L \times n}$ is the mean shape vector. An new shape instance can be generated as $s_p = \bar{s} + U_s \times p$, where $p = (p_1, p_2, \ldots, p_n)^T$ is the vector of shape parameters.

3.2 Motion Model

The motion model consists of a warp function **W(p)** which is essential for warping the texture related to a shape instance generated with parameters **p** into a common reference shape which is obtained from the automatic initialization process.

3.3 Appearance Model

The appearance model is trained by:

1. First extracting features from all the training images using the features function ***F()*** defined by holistic features, i.e., $F(I_i) \forall i \in 1,2,\ldots,N.$
2. Warping the feature-based images into the reference shape in order to get $F(I_i) W(p_i) \forall i = 1,2,\ldots,N.$
3. Vectorizing the warped images as $a_i = F(I_i) W(p_i) \forall i = 1,2,\ldots,N.$ where $a_i \in \Re^{M \times 1}$
4. Applying PCA on the acquired vectors which results in $(\bar{a}, U_a)$ where $U_a \in \Re^{M \times m}$ is the orthonormal basis of m eigenvectors and $\bar{a} \in \Re^{M \times 1}$ is the mean appearance vector.

A new appearance instance can be generated as $a_c = \bar{a} + U_a \times c$, where $c = (c_1, c_2, \ldots, c_n)^T$ is the vector of appearance parameters.

With an abuse of notation, let us define $t(W(p)) \equiv F(I) W(p)$ as the feature-based warped $M \times 1$ vector of an image I given its shape instance generated with parameters p

Many AAM versions differ on the way that this appearance warping ***W(p)*** is performed. I specifically used Holistic AAM

3.4 Holistic AAM

The HolisticAAM [12] uses a holistic appearance representation obtained by warping the texture into the reference frame with a non-linear warp function ***W(p)***. The warp function I employed is Thin Plate Splines. The reference frame is the mask of the mean shape's convex hull.

3.5 Cost Function and Optimization

Fitting an AAM on a test image involves the optimization of the following cost function

$$\arg\min_{p,c} (\| t W(p) - \bar{a} - U_a \times c \|)$$

with respect to the shape and appearance parameters. Note that this cost function is very similar to the one of Lucas-Kanade for Affine Image Alignment and Active Template Model for Deformabe Image Alignment [14].

The only difference has to do with the fact that an AAM aims to align the test image with a linear appearance model.

3.6 Lucas-Kanade Optimization [14]

The Lucas-Kanade optimization belongs to the family of gradient-descent algorithms. In general, the existing gradient descent optimization techniques are categorized as: (1) *forward* or *inverse* depending on the direction of the motion parameters estimation and (2) *additive* or *compositional* depending on the way the motion parameters are updated. I have used Wiberg Inverse-Compositional algorithm version which takes less computational power and is ideal for hand detection. It is a very efficient version of alternating optimization. It involves solving two different problems in an alternating manner, one for the shape and one for the appearance parameters increments. All these algorithms are iterative and the shape parameters are updated at each iteration in a compositional manner as

$$W(p) \leftarrow W(p) \circ (\Delta p)^{-1}$$

3.7 Wiberg Inverse-Compositional algorithm [15]

It involves solving two different problems in an alternating manner, one for the shape and one for the appearance parameters increments

$$\arg\min_{\Delta p} (\|Z\|)^2_{\hat{U}_a} \quad \text{where Z is} \quad tW(p) - \bar{a}(W(\Delta p)) - \sum c_i u_i W(\Delta p) \quad \text{.............(1)}$$

and

$$\arg\min_{\Delta c} (\|Y\|)^2 \quad \text{where Y is} \quad tW(p) - \bar{a}(W(\Delta p)) - \sum (c_i + \Delta c_i) u_i W(\Delta p) \quad \text{.........(2)}$$

Summation is from i=1 to m.

where $\hat{U}_a = I_e - U_a U_a^T$ is the "project-out" operator. Given the current estimate of Δc, the shape parameters increment estimated by solving the first optimization problem as

$$\Delta p = \hat{H}^{-1} \hat{J}_a^T (tW(p) - \bar{a})$$

where $\hat{J}_a = \hat{U}_a J_a$ is the projected-out Jacobian $J_a = (\nabla \bar{a})(\partial W / \partial p)_{p=0} + \sum c_{i=1,2,..m} \nabla u_i \times (\partial W / \partial p)_{p=0}$ and

$\hat{H}_a = \hat{J}_a^T \times \hat{J}_a$ being the Gauss-Newton approximation of the Hessian. Given the current estimate of **Δp**, the appearance parameters increment is computed by solving the second optimization problem as

$\Delta c = U_a^T [tW(p) - \bar{a} - U_a c - j_a \Delta p]$

The computational cost of Wiberg optimization is $\mathrm{O}((nm+n^2)M)$.

3.8 Learned Model The below figures show the predictive analysis of the learned AAM. The training set contains 32 manually annotated images. The parameters the are taken are the first four eigen vectors from PCA. I have used a total of 68 landmarks with 9 major landmarks excluding the reference point.

Figure20:

Figure21

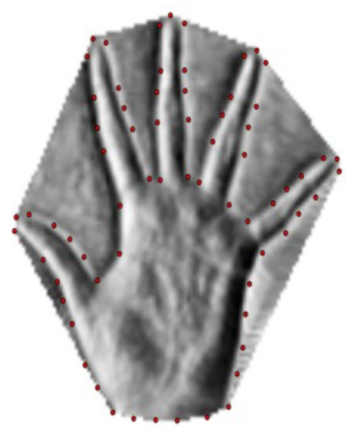

Figure22

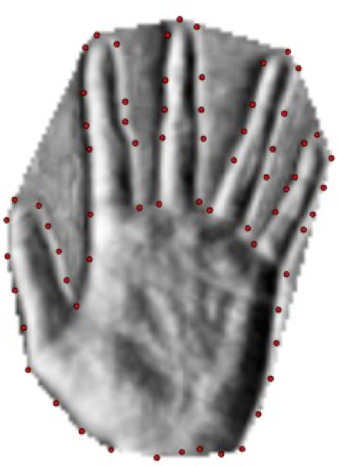

Figure23

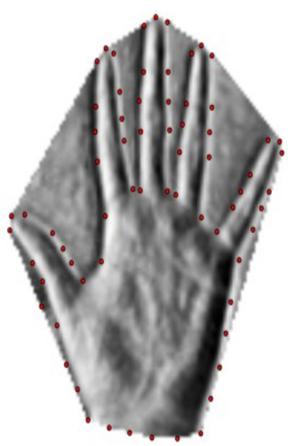

Figure24

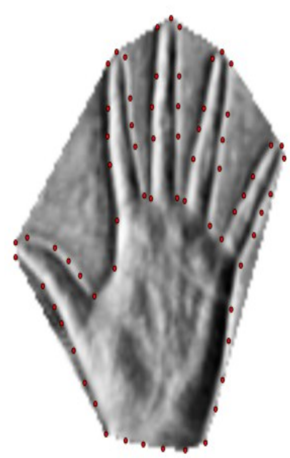

Figure25

Chapter 4

Algorithm for initialization

Now that the model is ready and region where the hand is present is know, the next step is to initialize the learned model. For this I propose an a novel algorithm for which the preprocessing steps and the corresponding results are shown below. The algorithm is given in 8^{th} step.

1)Skin segmentation

2)Filling the holes in the results obtained

3)Edge detection

4) Finding the largest connected contour

5)Finding the bottom most left point which will be the initial point using convex hull.

6)Following the contour from the initial point and landmarking all the points along the curve. Direction is noted in which the contour is being followed.

7)Noting all the points that are at Maximum and Minimum distances from the initial point.

8)Checking with the cases and applying the K-NN if necessary.

<u>Algorithm:</u>

case 1: All the 9 points are residing on a typical Distance map and are in domain of learned point cloud.

Automatically add other points between the major landmarks and initialize fitting.

case 2: All 9 are present but some or all are outside domain of the learned model points with respect to the initial point.

Use K- Nearest Neighbors (K-NN) [16] to find the closest point to the deviated point from the corresponding point cloud

case 3: Not more than 5 are missing and none or some or all are outside model domain

Proceed to case 2 for the points that re present and for the missing points choose the nearest neighbor for that landmark with respect to the initial points and the existing points.

case 4: More points are present and none or some or all are outside the model domain

Only landmark the points that are closest to the learned point cloud.

case 5: More than 5 are missing

It is very likely that wrong points might be localized. So, project the mean of the AAM on to the image and start fitting.

Note that a point is considered as a landmark only if the distance between that point and the previous landmark is greater than a predefined value in the distance map. Before calculating the distance the final contour is smoothed to avoid any local Minima or Maxima.

Chapter 5

Results

In this chapter results are discussed for the cases given is chapter 4.

Results of the preprocessing steps.

Test Image 1

Figure 26: skin segmentation Test Image 1

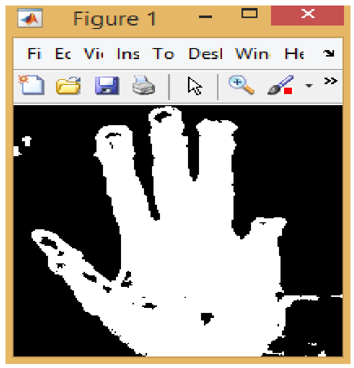

Figure 27: Binary segmentation

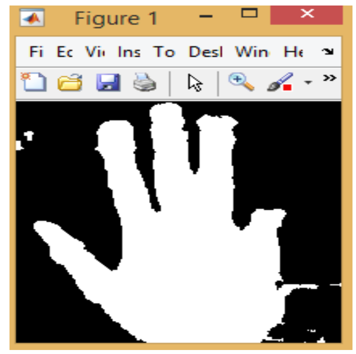

Figure 28: Filling holes in connected regions

Figure 29: Canny edge detection

Figure 30: Finding the largest connected contour

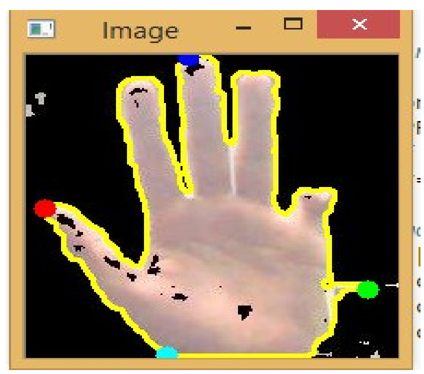

Figure 31: Extracting the bottom left point for initialization(Cyan)

Figure 32 shows the results without any modification and are obtained directly from the distance map. The 6^{th} landmark is far from the desired one and there is an extra point to the extreme right which is unnecessary. Figure 33 shows the corrected initialization where cases 2 and 4 are executed. Note that the 9^{th} landmark is deviated from the initial one as the algorithm looks for the closest neighbor in the trained point cloud.

Figure 32: Without Modification

Test Image 1

Figure 33: After using my algorithm

Test Image 1

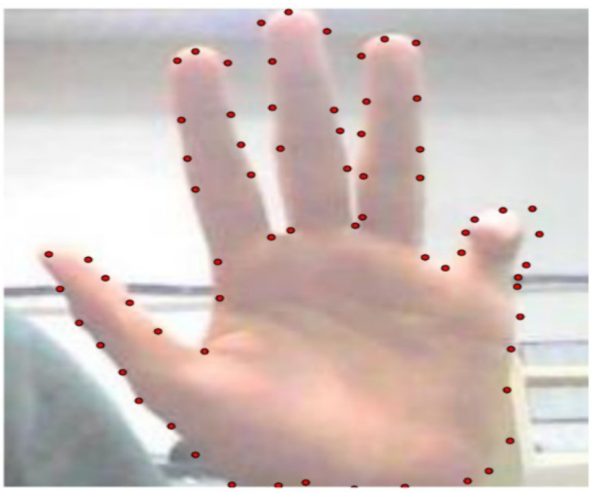

Figure 34: Final Segmentation Test Image 1

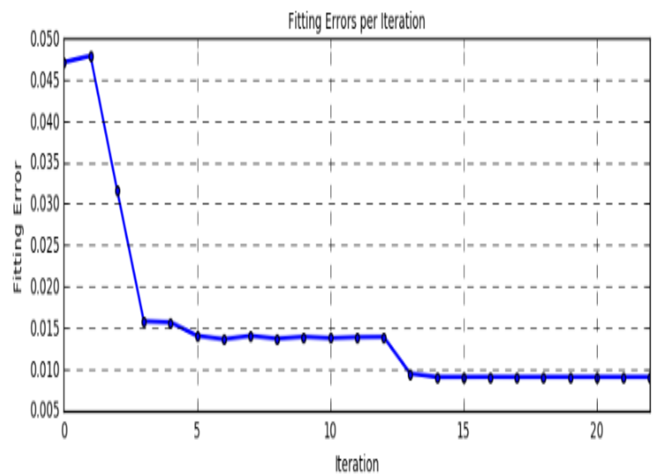

Figure35: Error plot 0.9 % error Test Image 1

$$\text{Error}= \frac{number\ of\ hand\ pixels\ missed + number\ of\ non-hand\ pixels\ segmented}{number\ of\ hand\ pixels}$$

Test Image 4

Now lets see the results where case 5 is executed.

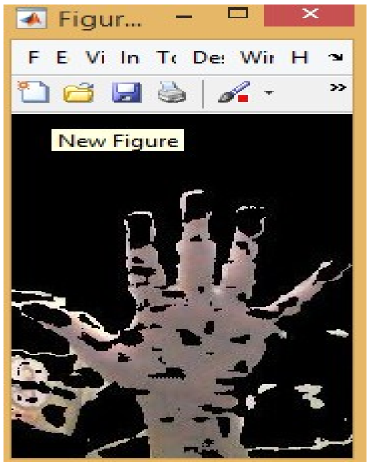

Figure36: Skin segmentation Test image 4

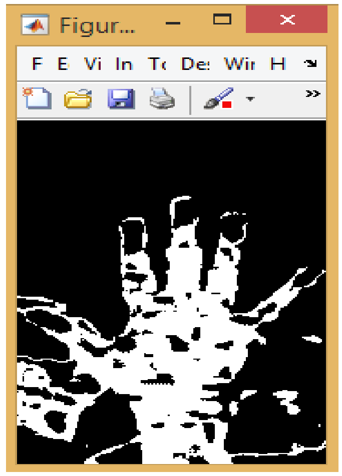

Figure37: Binary segmentation

Figure38: Filling regions

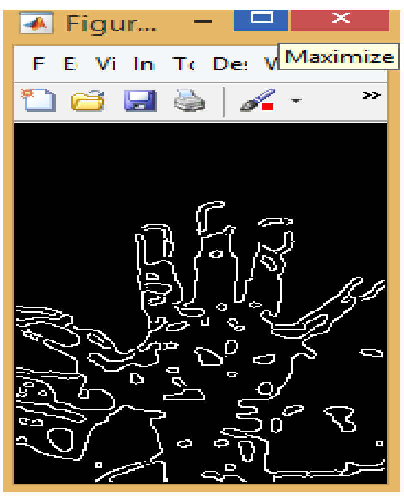

Figure39: canny edge detection

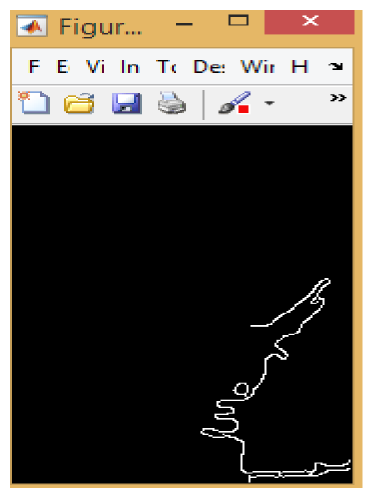

Figure40: Largest connected contour(More than 5 are missing)

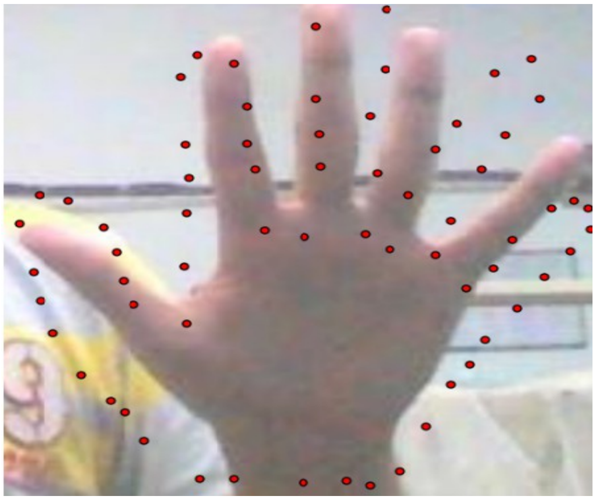

Figure 41: Initialization by using the mean model Test image 4

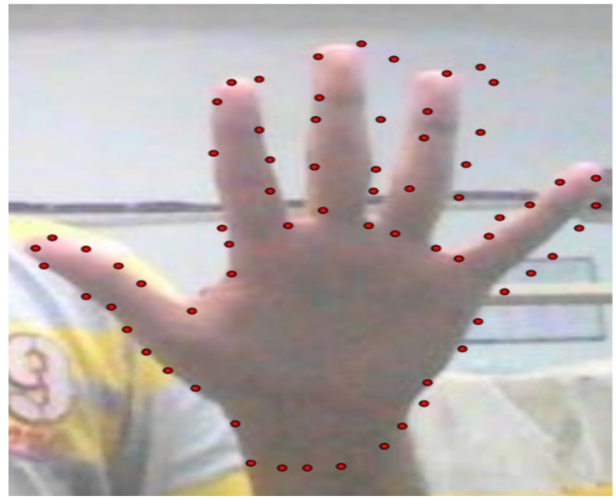

Figure 42: Final segmentation Test image 4

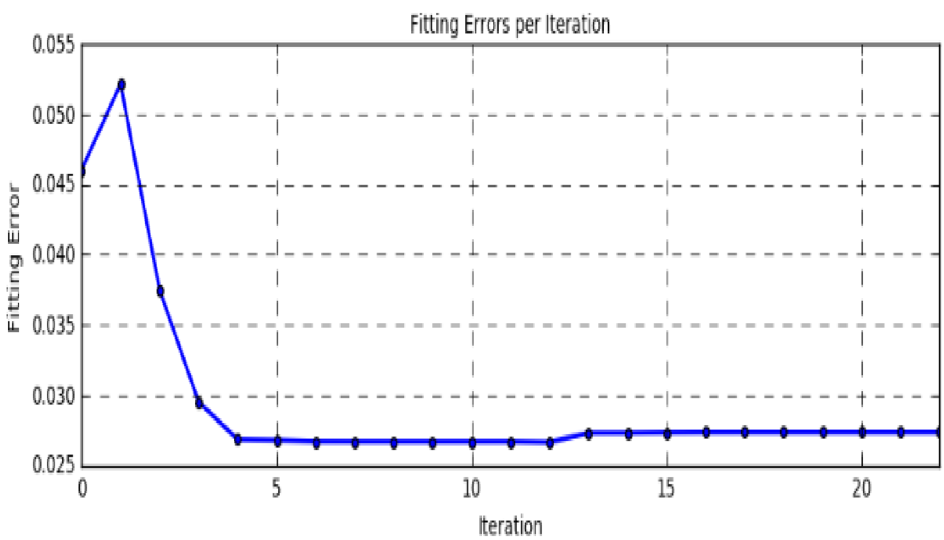

Figure 43: Error plot. 2.8 % Test 4

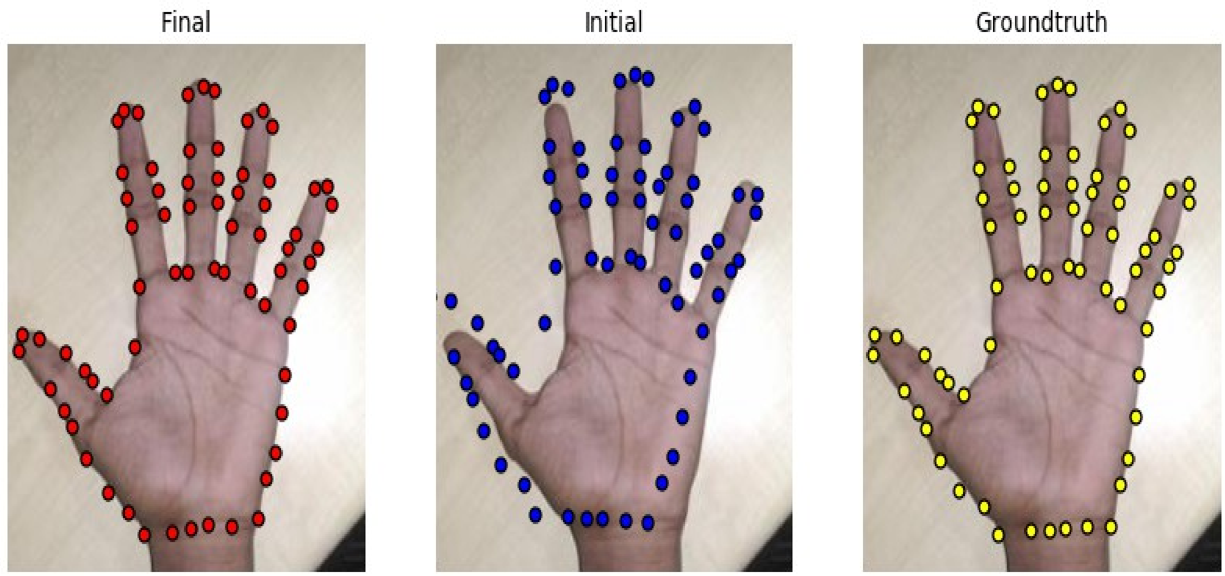

Figure 44: Best Result obtained from my database Test image 7

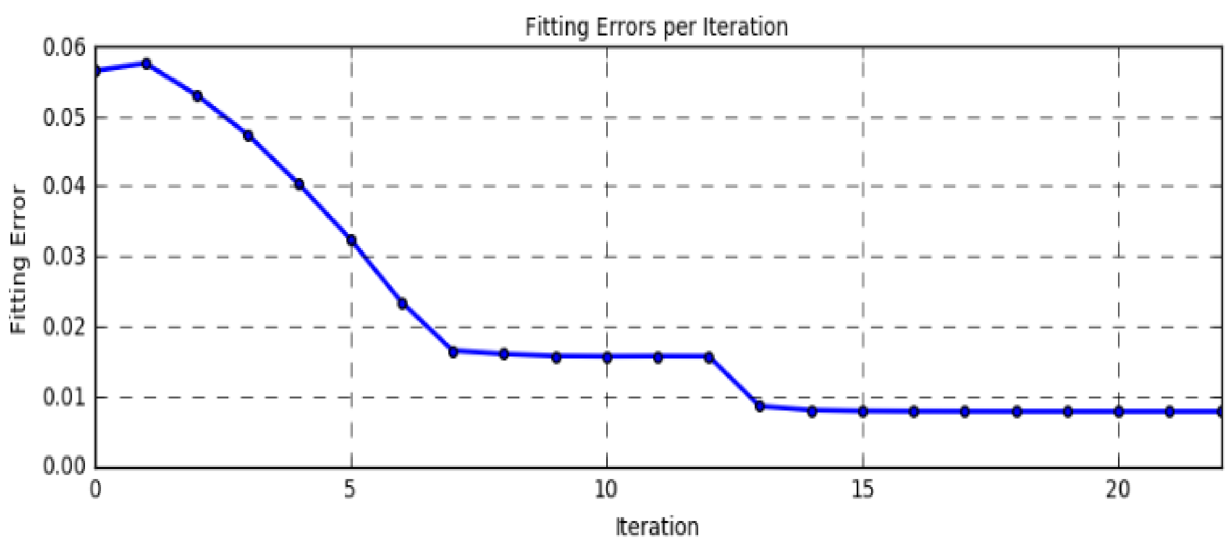

Figure 45: Error plot. 0.74% for Test image 7

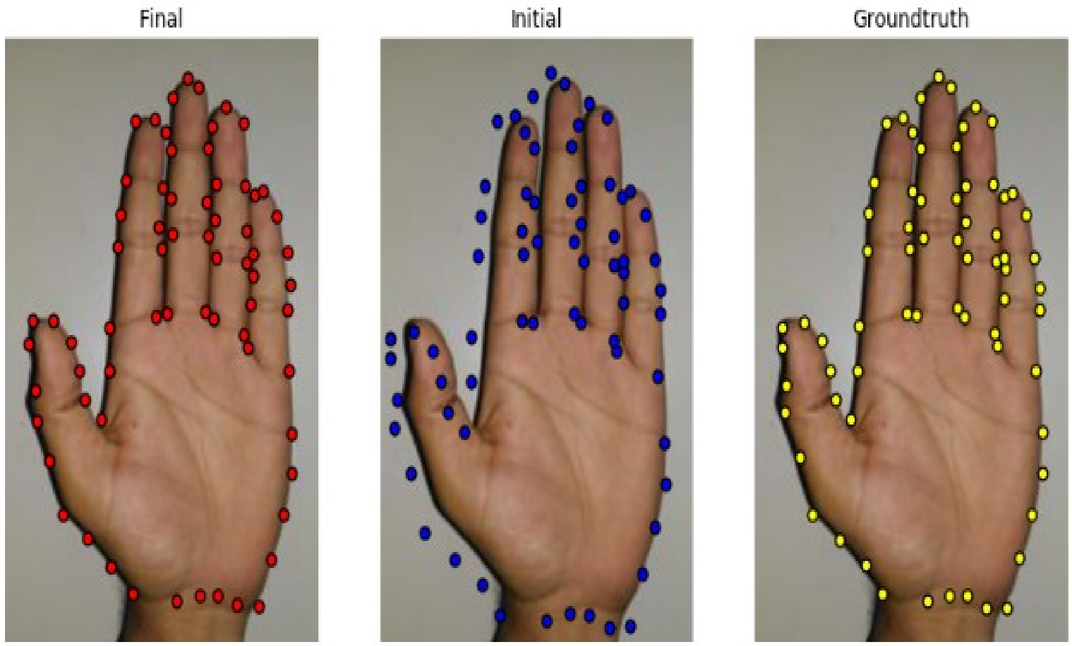

Figure 46: Case where the fingers are very close test image 8

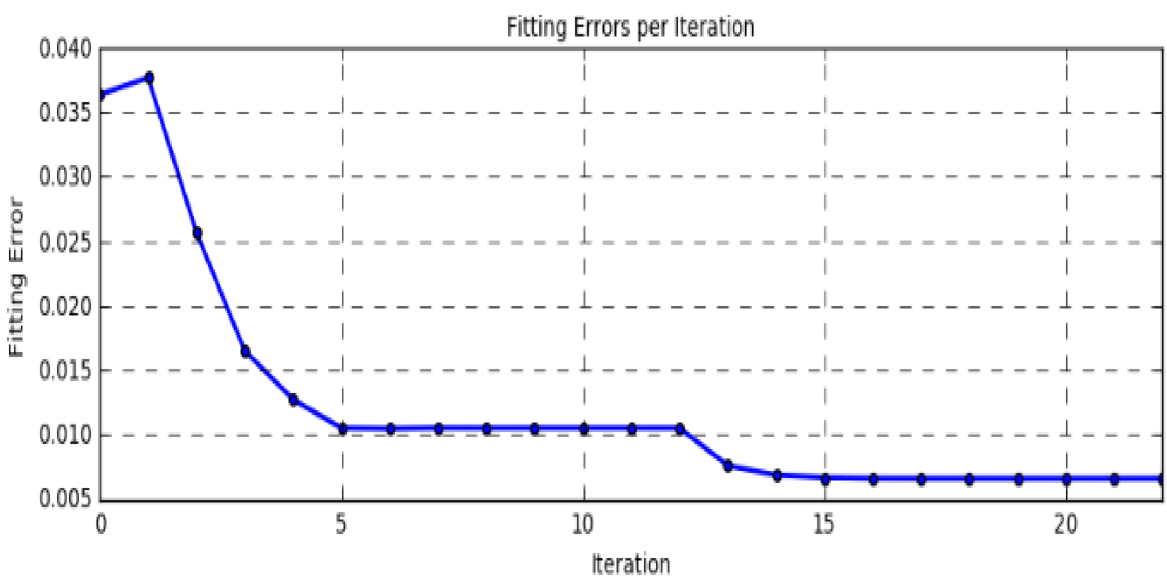

Figure 47: Error plot. 0.7% for test 8

Chapter 6

Contribution and limitations

6.1 Contribution

1. Hand detection is performed using a trained classifier rather than just skin color for better performances in cluttered backgrounds.

2. Used a modified version of AAM fitting.

3. Developed an algorithm to modify the existing distance map method for better initialization.

4. The resulting method is able to segment the hand from challenging backgrounds and illumination changes with the worst case error of 2.8% from the database.

6.2 Limitations

1. Haar cascade classifier giving on average of 3 false detections.

2. My initialization algorithm for improving the initialization doesn't work if more than 5 initial locations are missed and if the final contour has several maxima and minima close to each other. In that case Mean Model is projected directly without initialization.

Chapter 7

Conclusion and Future work

1. By using both shape and appearance features of hand better segmentation results are achieved even if the initialization is not perfect.

2. A novel algorithm is proposed to modify the initialization using distance map.

3. Detection phase can be improved by using the method proposed at [9].

4. Skin color segmentation performance can be improved using more recent deep learning methods(CNN).

Chapter 8

References

[1] T.F. Cootes and C.J. Taylor, "Statistical models of appearance for computer vision", Technical report, University of Manchester, United Kingdom, 1999.

[2] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of Adjacent Contour Segments for Object Detection", PAMI, 2007.

[3] J. Doublet, O. Lepetit and M. Revenu, "Hand detection for contactless biometrics identification", Cognitive System with Interactive Sensors, Paris, 2006.

[4] J. Alabort-i-Medina and S. Zafeiriou, "A Unified Framework for Compositional Fitting of Active Appearance Models", arXiv:1601.00199.

[5] J. Alabort-i-Medina and S. Zafeiriou, "Bayesian Active Appearance Models", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[6] J. Alabort-i-Medina, and S. Zafeiriou, "Unifying Holistic and Parts-Based Deformable Model Fitting", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[7] T.F. Cootes, G.J. Edwards, and C.J. Taylor. "Active Appearance Models", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 6, pp. 681–685, 2001.

[8] T.F. Cootes and C.J. Taylor. "Active shape models-'smart snakes", British Machine Vision Conference (BMVC), 1992.

[9] A. Mittal, A. Zisserman, and P. H. S. Torr, "Hand detection using multiple proposals", In British Machine Vision Conference, 2011.

[10] P. Viola and Michael, "Rapid Object Detection using a Boosted Cascade of Simple Features".

[11] Freund, Yoav, Schapire and E. Robert, "A decision-theoretic generalization of on-line learning and an application to boosting".Journal of Computer and System Sciences, 1997.

[12] J. Alabort-i-Medina and S. Zafeiriou, "A Unified Framework for Compositional Fitting of Active Appearance Models".

[13] T.F. Cootes, G. Edwards and C.J. Taylor, " Comparing Active Shape Models with Active Appearance Models ".

[14] B. D. Lucas and T. Kanade , "An iterative image registration technique with an application to stereo vision".

[15] G. Papandreou, and P. Maragos, "Adaptive and constrained algorithms for inverse compositional active appearance model fitting", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008.

[16] N. S Altman, "An introduction to kernel and nearest-neighbor nonparametric regression", The American Statistician, 1992.